Final Report for SBIR N02-190
Efficient Numerical Methods for Stable Distributions
Contract N00421-03-P-0059
Report SBIR-7

John P. Nolan
Robust Analysis, Inc.
6618 Allegheny Avenue
Takoma Park, MD 20912

12 June 2003

1 Work Performed and Results Obtained

The objectives of the contract were to develop computational methods for stable distributions. This section will describe the work performed and the results obtained, organized by topics in approximate chronological order. The second section discusses technical feasibility of Phase II work, and a short third section discusses miscellaneous issues.

We note that there is more extensive documentation on these topics in the monthly reports and delivered software. The six monthly reports, the extensive report on simulations evaluating parameter estimation methods, and the user manual for the STABLE matlab interface totaled over 200 pages of detailed information. The information below is a summary of that work, it is not intended as a complete record.

There are multiple parameterizations for stable laws and much confusion has been caused by these different parameterizations. We define two parameterizations below, at least six more have appeared in print. In most of the recent literature, the notation $S_\alpha(\sigma, \beta, \mu)$ is used for the class of stable laws. We will use a modified notation of the form $S(\alpha, \beta, \gamma, \delta; k)$ for three reasons. First, the usual notation singles out α as different and fixed. In statistical applications, all four parameters $(\alpha, \beta, \gamma, \delta)$ are unknown and need to be estimated; the new notation emphasizes

Report Documentation Page

Report Date	Report Type	Dates Covered (from... to)
0/06/2003	N/A	-

Title and Subtitle Efficient Methods for Stable Distributions	**Contract Number** N00421-03-P-0059
	Grant Number
	Program Element Number
Author(s)	**Project Number**
	Task Number
	Work Unit Number
Performing Organization Name(s) and Address(es) Robust Analysis Inc., 6618 Allegheny Avenue, Takoma Park, MD 20912	**Performing Organization Report Number**
Sponsoring/Monitoring Agency Name(s) and Address(es)	**Sponsor/Monitor's Acronym(s)**
	Sponsor/Monitor's Report Number(s)
Distribution/Availability Statement Approved for public release, distribution unlimited	
Supplementary Notes	
Abstract	
Subject Terms	
Report Classification unclassified	**Classification of this page** unclassified
Classification of Abstract unclassified	**Limitation of Abstract** UU
Number of Pages 16	

this. Second, the scale parameter is not the standard deviation (even in the Gaussian case), and the location parameter is not generally the mean. So we use the neutral symbols γ for the scale (not σ) and δ for the location (not μ). And third, there should be a clear distinction between the different parameterizations; the integer k does that explicitly.

The first parameterization is continuous in all four parameters and is used in all our statistical work. The second parameterization is the one used in most recent publications it is simpler to use for theoretical work. It has a discontinuity as $\alpha \to 1$, which makes it impractical for numerical and estimation purposes.

A random variable X is $S(\alpha, \beta, \gamma, \delta; 0)$ if it has characteristic function

$$E \exp(iuX) = \begin{cases} \exp\left(-\gamma^\alpha |u|^\alpha \left[1 + i\beta(\tan\frac{\pi\alpha}{2})(\operatorname{sign} u)(|\gamma u|^{1-\alpha} - 1)\right] + i\delta u\right) & \alpha \neq 1 \\ \exp\left(-\gamma|u|\left[1 + i\beta\frac{2}{\pi}(\operatorname{sign} u)\ln(\gamma|u|)\right] + i\delta u\right) & \alpha = 1. \end{cases}$$

A random variable X is $S(\alpha, \beta, \gamma, \delta; 1)$ if it has characteristic function

$$E \exp(iuX) = \begin{cases} \exp\left(-\gamma^\alpha |u|^\alpha \left[1 - i\beta(\tan\frac{\pi\alpha}{2})(\operatorname{sign} u)\right] + i\delta u\right) & \alpha \neq 1 \\ \exp\left(-\gamma|u|\left[1 + i\beta\frac{2}{\pi}(\operatorname{sign} u)\ln|u|\right] + i\delta u\right) & \alpha = 1. \end{cases}$$

1.1 Discrete Stable Distributions

Given a continuous stable distribution $Z \sim S(\alpha, \beta, \gamma, \delta; k)$, and two cutoff values $c_1 < c_2$, the quantized and truncated r.v.

$$X = \begin{cases} c_1 & Z \leq c_1 + 1/2 \\ \operatorname{round}(Z) & c_1 + 1/2 < Z < c_2 - 1/2 \\ c_2 & Z \geq c_2 - 1/2 \end{cases}$$

is called a discrete stable distribution. In the examples, parameterization $k = 0$ and cutoffs $c_1 = -128$ and $c_2 = +127$ are used, corresponding to the common values used in digital signal processing.

Five new functions for discrete/quantized stable distributions were written.

- `sgendiscrete` generates discrete stable random variates. It works by generating continuous stable random variables using the Chambers-Mallows-Stuck method, rounding them to the nearest integer, and then cutting off if the value is too high or too low.

- `spdfdiscrete` computes the pdf f_{disc} of a discrete stable distribution. This works by computing the probability that the continuous random variable is in the interval of width one centered at the current value: $P(X = x) = P(x - 1/2 < Z \leq x + 1/2)$.

- `scdfdiscrete` computes the cdf F_{disc} of a discrete stable distribution. It works in a similar way: $P(X \leq x) = P(Z \leq x + 1/2)$. (Both the latter two functions have special conditions for $x \leq -128$ and $x \geq 127$.)

- `sdiscretefindgamma` computes the value of the scale function γ needed to achieve a certain saturation probability $p_{sat} = P(X = c_1) + P(X = c_2)$. This is used in cases where a certain saturation probability, say 0.02, is needed.

- `sgendiscrete2` generates discrete stable random variates. It is similar to `sgendiscrete`, but instead of requiring the scale parameter γ, it uses a user specified saturation probability to determine the scale (using the previous function) and then generates the random variates.

Figure 1.1 shows the output of these functions for one example: the top graph is a histogram of 10000 simulated random variates, the second graph shows the exact pdf, and the third graph shows the exact cdf for the quantized/discrete distribution. The two key differences between this and the continuous case are that only integer values in the range -128 to +127 are observed and that the truncation or saturation causes a certain probability to be concentrated at the endpoints.

1.2 matlab Interface

In the second month of the contract, a matlab interface for the STABLE functions was developed. The software was delivered to the Navy in late December. It consists of

- stablemex.dll - a dynamic link library that contains the core functions of STABLE, including the new discrete stable functions. For efficiency reasons, these routines are coded in Fortran.

- matlab .m functions that define the matlab interface to stablemex.dll

- a user manual

- miscellaneous instructions and help files

The interface was updated later to incorporate the new methods described below. This interface represents a significant advance in the availability of tools for working with stable distributions. It allows engineers and scientists to analyze data and work with stable distributions within the common matlab environment they use. We comment briefly on the commercialization of this in the last section.

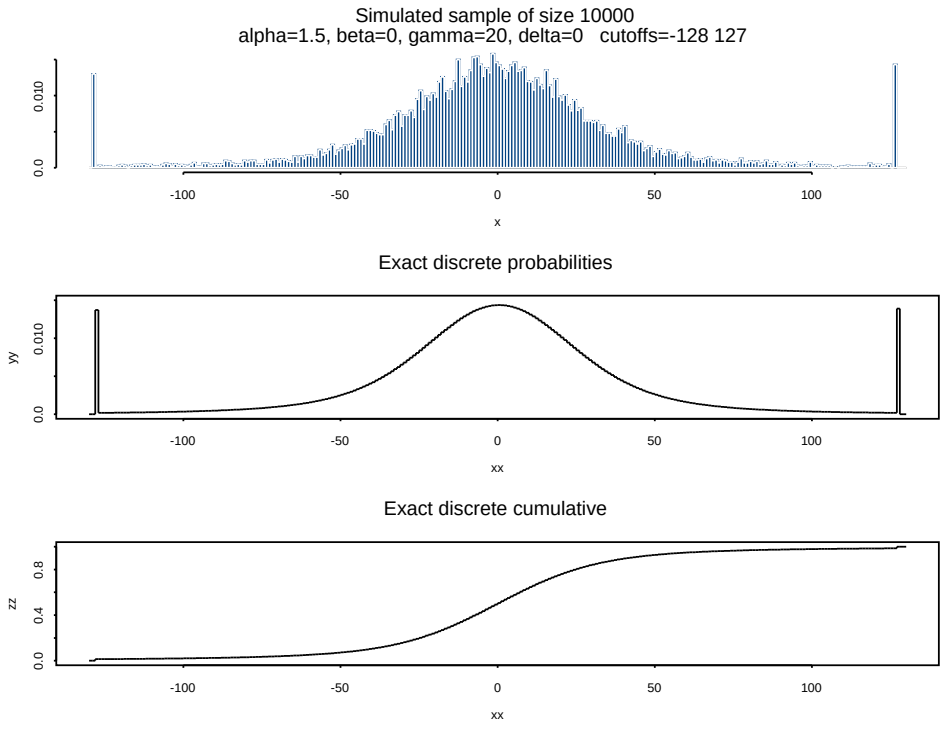

Figure 1: A centered discrete stable distribution with $(\alpha, \beta, \gamma, \delta)$ as shown.

1.3 Parameter Estimation

For a given data set $X = (x_1, \ldots, x_n)$ of independent discrete, truncated stable random variables, the underlying stable parameters can be estimated by using each of the following methods, with references to their published descriptions.

- **Quantile based estimation** Fama and Roll (1968), McCulloch (1986) and Ojeda (2001).
- **Characteristic function based estimation (char.fn.)** Koutrouvelis (1980) and Kogon and Williams (1998).
- **Maximum likelihood estimation (mle)** Nolan (2000).
- **Discrete maximum likelihood estimation (dmle)**

The first three methods are techniques were developed for continuous stable data, without discretization and truncation. The fourth one is a new technique, developed under this contract, where we explicitly take into account the discrete nature of the data and the fact that it is truncated. Given values of α, β, γ and δ, the likelihood of a discrete data set with known cutoffs c_1, c_2 is

$$L(\alpha, \beta, \gamma, \delta | X, c_1, c_2) = \prod_{i=1}^{n} f_{disc}(x_i | \alpha, \beta, \gamma, \delta, c_1, c_2),$$

where $f_{disc}(x| \ldots)$ is the probability density (mass function) of a discrete, truncated stable distribution. This likelihood can be numerically computed using the program developed in earlier in this contract to evaluate $f_{disc}(x| \ldots)$ for a discrete stable distribution. This likelihood is then numerically maximized in 4-dimensions using a multivariate optimization routine. The first three methods can be applied to discrete data, but will not work well in many cases. The next section describes the evaluation of these methods.

The matlab interface includes the first three methods and was extended to include the last method. It is:

- stablefitdmle to compute the discrete maximum likelihood estimators for a discrete stable fit to integer valued/truncated data.

Implementing this routine took longer than expected. The "typical" case worked well, but there were problems at certain values of the parameters: near the Lévy case ($\alpha = 1/2$, $\beta = \pm 1$), near $\alpha = 1$, and near the Gaussian case ($1.99 < \alpha$). Specifically, if α and β were in one of these special regions, the STABLE routine that computed the likelihood rounded the parameter values to nearby values to

avoid computational difficulties. This had the hidden effect of making the likelihood surface look flat near those regions, which caused the optimization routine to get trapped in the region. These problems were fixed in late May.

1.4 Evaluation of Estimation Methods

A large scale simulation and evaluation of estimation procedures for discrete stable distributions was done and a long (173 page) report was filed in month 5. We note that Ojeda (2001) showed that for continuous data, the quantile method is least efficient and the maximum likelihood method is most efficient. The characteristic function is intermediate, in fact almost as efficient as maximum likelihood. This is not the case for discrete stable data. We will not repeat the results of the month 5 report here, only give a brief summary of what we found for typical values of the parameters.

- **Quantile based estimation** worked reasonably well when the data was not too saturated. For reasonable saturations, say less than 5% on either tail, the quantile method is unaffected by the masses at the cutoff points, as it only uses the 5-th, 25-th, 50-th, 75-th and 95-th percentiles of the data. The discretization causes some rounding in the estimation of these quantiles, and hence some inaccuracy in parameter estimates. When compared to the second and third methods, this method is less efficient for continuous data, see Ojeda (2001), but generally works better than either for discrete data.

- **Characteristic function based estimation** works well for continuous data, but poorly for discrete data. As in the mle case, the problem is in the truncation. This method works by estimating the sample characteristic function/Fourier transform. The shape of this function near the origin is used to estimate parameters. The values of the sample characteristic function near the origin are determined by large values of the data. The truncation eliminates those large values, and makes the characteristic function estimator perform poorly for truncated data in almost all cases.

- **Maximum likelihood estimation** did not seem to be much affected by rounding (quantization), but could be significantly affected by the truncation. Roughly speaking, the chopping off of the tails resulted in a "light tailed" data set, which this method regularly estimated as Gaussian ($\alpha = 2$).

- **Discrete maximum likelihood estimation** generally works best. This is as expected, because the method explicitly models the exact model for the data, taking into account the discretization and truncation. It is slower than the

other methods, due do the computationally intensive nature of the process. The symmetric case is faster, as it uses the fast approximation to the continuous cdf described below.

For illustration purposes, we show results of one simulation. The test case we examined used $\alpha_0 = 1.5$, $\beta_0 = 0$, $\delta_0 = 0$ and varying saturation probability $p_{\rm sat}$. $M = 100$ i.i.d. samples of size $n = 1,000$ were generated and each parameter was estimated using each of the methods. The results were saved to a file and then analyzed using a separate program. The analysis computed the means squared error (MSE) of each parameter, e.g. MSE$(\alpha) = (1/M)\sum_{i=1}^{M}(\hat{\alpha}_i - \alpha_0)^2$, where $\hat{\alpha}_i$ is the estimate of α from the $i^{\rm th}$ sample. Figure 2 shows the plots of the MSEs for each parameter and each of the estimation methods.

There are a few cases where parameter estimation will not work well for discrete stable distributions. If the granularity is too coarse, all the data will be clumped at a few values. Likewise, if the saturation probability is very high, then the distribution is close to two large point masses at -128 and +127. In either case, it is not possible to recover parameters. There were two cases where the simulations in month 5 showed poor performance of the DMLE method. When α was near 2 or when α was near 1/2, the truncated tail resulted in some initial estimates of α being 2 and the computational problem discussed above caused the DMLE algorithm to get trapped at $\alpha = 2$. These initial estimates caused an artificially large MSE for the DMLE algorithm. That problem has been fixed and the performance of the DMLE method is much improved in these cases.

1.5 Fast Approximation of Densities and Cumulatives

The STABLE routines for computing F and f, scdf and spdf, use numerical evaluation of certain integrals, see Zolotarev (1986) and Nolan (1997). The numerical integration gets about 12 digits of absolute significance. These routines have a few weaknesses: they are slow when doing intensive computations (e.g. numerical maximum likelihood estimation), when computing the cdf or pdf on the extreme tails, and on the light tail, e.g. $x < 0$ when $\beta = +1$, the quantities get very small ($< 10^{-12}$) and are hard to accurately estimate. Also, as α gets smaller, the stable pdf gets extremely peaked as $\alpha \downarrow 0$, yet also have very heavy tails. It is very hard to accurately compute the cdf F and pdf f for small α.

We will focus on the cdf in what follows. Our approach uses the transformation

$$Y = X^{<\alpha>} := (\text{sign}X)|X|^{\alpha}$$

to handle the computational problems when α is small. Let $G(y|\alpha, \beta; 0)$ denote the cdf of Y, when X has cdf $F(x|\alpha, \beta; 0)$. The first nice property of the signed

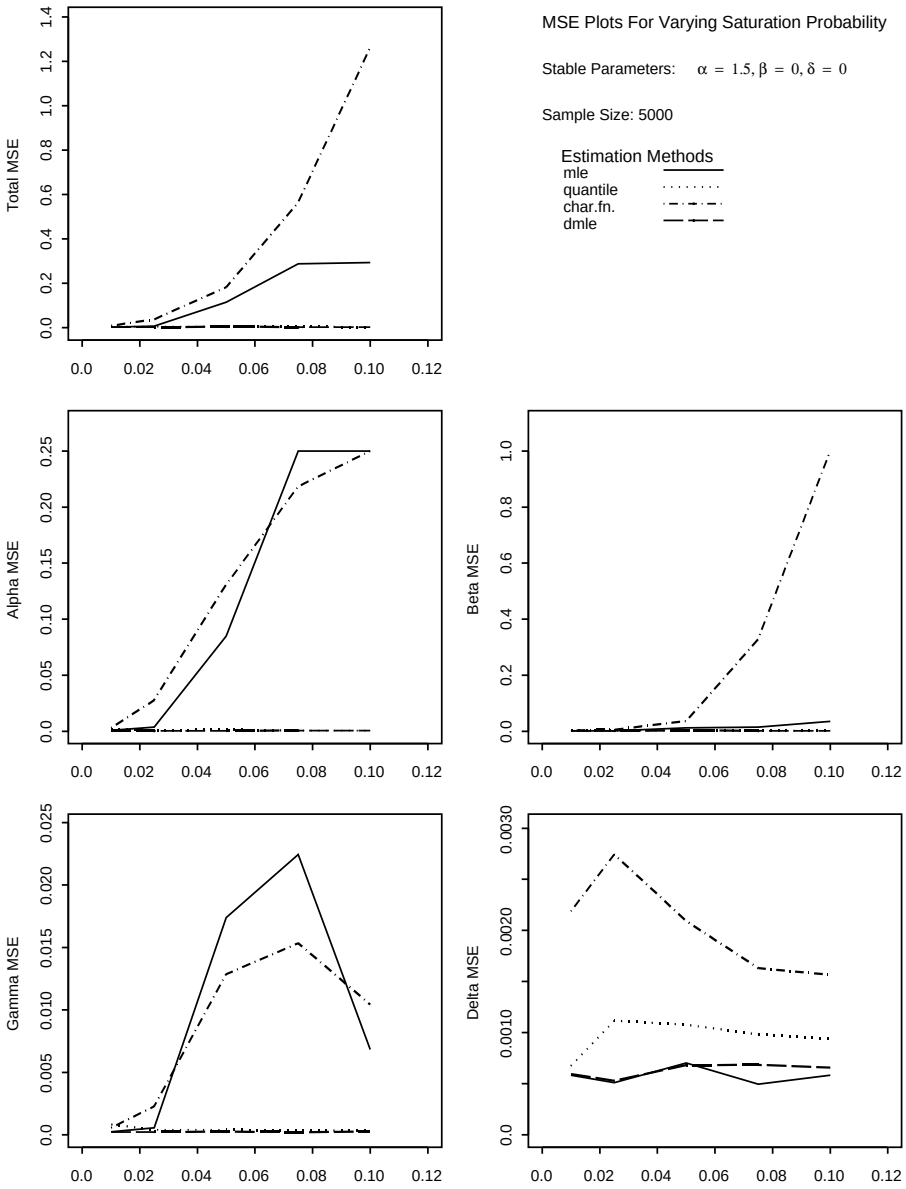

Figure 2: MSE plots from one simulation run.

power transformation is that it is 1-to-1 and easily invertible: $y = x^{<\alpha>}$ if and only if $x = y^{<1/\alpha>}$, i.e. if we know $G(\cdot)$, then we can easily compute $F(\cdot)$. Second, in the symmetric case, we only need to consider $y \geq 0$. Third, the tail probabilities all behave the same now: for $y >> 1$,

$$P(Y > y) = P(X > y^{1/\alpha}) \sim c_\alpha y^{-1},$$

hence the tail approximation becomes very simple. Finally, it can be shown that as $\alpha \downarrow 0$, $G(y|\alpha, \beta; 0) \rightarrow G(y|0, \beta; 0)$, where the right-hand side is a smooth, proper distribution function. This means that the degenerate behavior seen in $F(x|\alpha, \beta; 0)$ as $\alpha \downarrow 0$ is eliminated by the transformation. These features of G make it a useful tool for developing a compact, fast, and accurate approximation to F for all values of α.

After the $Y = X^{<\alpha>}$ transformation we use another 1-1 transformation T_α that is smooth and explicitly invertible, and that makes

$$H(y, \alpha, \beta) := P(\, T_\alpha(Y) \leq y\,)$$

relatively smooth. We then evaluate $H(y, \alpha, \beta)$ on a grid of y, α, β values, compute a spline interpolant for H, and save the spline coefficients. Then the cdf of X can be recovered by

$$F(x|\alpha, \beta) = H(T_\alpha^{-1}(x^{<\alpha>}), \alpha, \beta).$$

The choice of T_α is a technical issue, and relates to the goal of having a fast and accurate approximation over the range of the parameter space. In general there are four cases where approximations are difficult: (i) α small, (ii) α near 1 in the non-symmetric case, (iii) α near 2, (iv) β near ± 1. The transformation $Y = X^{<\alpha>}$ works well on the first problem, and using the continuous 0-parameterization works well on the second problem. However, we have not been able to find a general solution to the third and fourth problems.

We have developed a quick approximation method for the important special case of symmetric stable cdfs. It has a reasonable work-around for the third problem above, and the symmetry avoids the last problem. It uses $T_\alpha(y) = \log y$, computes $H(y, \alpha, 0)$ for $\alpha \in [0, 1.999]$, and uses a 2-dimensional spline to approximate the values of H everywhere. The code is implemented in Fortran subroutine QKSCDFSYM (QuicK Stable CDF SYMmetric) and speeds up the evaluation of symmetric stable cdfs by a factor of over 200.

The accuracy of this approximation is quite good, except as α approaches 2. It turns out that the signed power transformation eliminates most of the problems for small alpha, but aggravates problems for large α. We planned to use duality of stable laws with $\alpha > 1$ to stable laws with $\alpha < 1$ to eliminate the need to

consider large α. However, the problem is complicated by the fact that the duality relation requires the use of non-symmetric stable cdfs with $\alpha < 1$ to compute the symmetric stable cdfs with $\alpha > 1$. We have tried more than 10 transformations T to deal with the nonsymmetric case, but have not been to handle it in a satisfactory way yet.

We summarize the existing fast functions in STABLE:

- qkscdfsym - a fast approximation to symmetric stable cdfs for all $\alpha \in (0, 1.999]$
- qkscdf - slower approximation to the stable cdf in the general (skewed) case. It uses the numerical integration formulas to compute the cdf on a grid of x values, and fits a spline to these values. This setup process takes some time, but if there are a large number of x values at which to evaluate the cdf, this method is faster than evaluating each by numerical integration.
- qkspdf - a fast approximation to all stable pdfs with $\alpha \in [0.4, 1.99]$. It uses a precomputed 3-dimensional spline approximation, with no numerical integration. It is fast, but requires a lot of storage (over 20,000 lines of precomputed constants).

1.6 Nonlinear Function

The nonlinear function, also called the score function for the location, is

$$g(x) = -\frac{f'(x)}{f(x)}.$$

This function is used in the locally most powerful detector. We have developed four methods of computing it, called g_0, g_1, g_2 and g_3 below.

1.6.1 g_0 numerical evaluation using pdf

The first method of evaluating g uses the numerical quadrature routines to evaluate the pdf $f(x)$. It then uses Ridder's method to numerically find the derivative of the pdf at a point. It is accurate, but slow; we use it as our baseline method. It typically takes $\sim$16 pdf evaluations to estimate the derivative, so it typically takes $\sim$17 numerical quadratures to evaluate $f'(x)/f(x)$ at a single x.

1.6.2 g_1 numerical evaluation using approximation to the pdf

The first improvement in calculating $g(x)$ is to replace the the pdf routine used in the calculation of $g_0(x)$ by the faster approximation for the pdf (qkspdf). The

resulting function we will call $g_1(x)$. The same number of pdf calls are made, but when $g(x)$ is evaluated at a large number of points, the evaluation of $f(x)$, and hence $f'(x)$, are much faster. There is some cost to setting up the approximation, so this routine is only useful if evaluating $g(x)$ at a moderate or large number of x's.

1.6.3 g_2 spline approximation

The next method of approximation uses a spline interpolant to the nonlinear function. It uses $g_0(x)$ to compute the nonlinear function on a grid of points, then fits a shape preserving spline to those points. An ad hoc tail approximation is used to approximate $g(x)$ outside of the range of grid points. This approximation is called $g_2(x)$.

For g_1, a spline was fit to the pdf $f(x)$, and then that was used to numerically compute $g(x)$ at each x. In contrast, this method fits a spline directly to $g(x)$, so after the setup cost, evaluating $g_2(x)$ involves some searching for the correct interval, and then computing a cubic polynomial.

Currently the grid consists of 101 evenly spaced points on the interval [-20,20]. In addition to evaluating g_0 at these 101 points, there is a noticeable setup cost of computing the spline interpolant. For a large number of x's, the resulting approximation is faster than either g_0 or g_1.

1.6.4 g_3 rational approximation

The last approach uses a rational approximation to $g(x)$ of the form

$$g_3(x) = \frac{x + c_1 x^3}{c_2 + c_3 x^2 + c_4 x^4}.$$

It was motivated by the fact that in the symmetric case, $g(x)$ is odd ($g(-x) = -g(x)$), is linear near the origin and has tail behavior $g(x) \sim (1+\alpha)/x$ as $x \to \pm\infty$. A rational function of the above form has this behavior and can do a reasonable job of approximating $g(x)$ in the symmetric case when $\alpha > 1$.

The constants depend on the value of α; simple expressions were given for them in the report for month 6. This approximation is much faster than any of the above methods. It requires no calculations of the pdf, no computing of spline interpolants, no numerical derivatives, no interval searching, and almost no storage or code requirements. For a given α, the coefficients are computed once and $g_3(x)$ is evaluated directly for each x. Algebraic rearrangement of the formula for $g_3(x)$ makes it possible to evaluate it with 3 additions, 6 multiplications and 1 division. Finally, because evaluating g_0, g_1 and g_2 require many complicated operations,

	seconds	time(g_0)/time(g_k)
g_0	182.672	1
g_1	0.390	467
g_2	84.032	2.2
g_3	0.031	5870

Table 1: Timing results for g function approximations, $n = 20,000$ evaluations.

there is no guarantee that they behave well at all points. In contrast, because of the simple form of g_3, it is always well behaved.

We repeat that this method is good for $\alpha > 1$ and β near 0. Since this region is what seems to be of most interest in engineering applications, we believe this method is valuable even though the range is limited. We tried to adapt this approach to the nonsymmetric case, but found it difficult to get a reasonable method that works well for all α and β.

Figure 3 shows plots of the various functions in the symmetric case. Table 1 shows the results of a test run to determine the time required by the different methods by evaluating g_k, $k = 0, 1, 2, 3$ at n =20,000 x values. The time taken naturally depends on the computer used, so relative times (column 3) should be considered, not the absolute times (column 2). Also note that the times for g_0 and g_3 should be of the form $c_k n$. In contrast, g_1 and g_2 times include the initial setup time; hence the time for these two are of the form (setup time)$_k + c_k n$. If n is larger, the time per function evaluation decreases.

- stablenonlinfn computes g_0
- stableqknonlinfn to approximate the approximations to the nonlinear function g_1, g_2 and g_3 as described above. A parameter indicates which method to use.

We note that all of these routines behave poorly when α is small, say $\alpha < 1/2$ or when β is near +1 or -1. In the first case, the pdf varies very quickly and it is difficult to calculate $f(x)$ and even harder to accurately calculate $f'(x)$. When β is ± 1, the light tail decays much faster than in all other cases. In the case of the Lévy distribution, $g(x)$ has a vertical asymptote; similar behavior appears to occur for all $\alpha < 1$. For $\alpha \geq 1$, the nonlinear function is also poorly behaved when β is near ± 1.

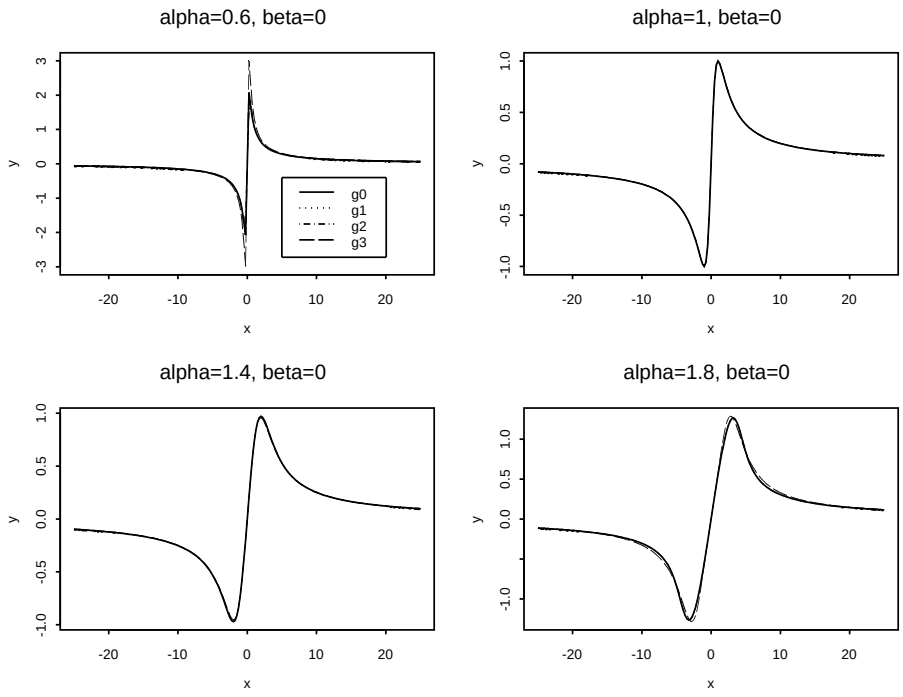

Figure 3: Comparison of the different methods of calculating $g(x)$ when $\beta = 0$.

1.7 Graphical Diagnostics and Test Script

The following matlab functions have been provided for visual assessment of a stable fit and for testing the STABLE matlab interface. They use the core STABLE functions.

- stabledensityplot to produce a diagnostic plot showing smoothed data density and the density of a stable fit
- stableqqplot to produce a diagnostic plot showing the quantiles of the stable fit and the data
- stablepppplot to produce a diagnostic plot showing the cdf of the fit and the cdf of the data
- stablezzplot to produce a diagnostic plot showing Φ^{-1} (the inverse of the normal cdf) applied to a both axes of a pp-plot
- stablediag to produce all four of the above
- stabletest to test most of the functions in STABLE from matlab. It is also a source of examples on how to use the different routines.

2 Technical Feasibility

The results of this Phase I contract show that it is possible to develop computationally efficient methods for working with stable distributions. There are now accurate and reliable routines to compute densities, cumulative distribution functions and many other quantities quickly and accurately. And there have been simulations to evaluate the estimation routines and we have a solid understanding of what works and what does not.

There are still one dependency on the IMSL routines in our Fortran routines. The one case that we have not been able to eliminate the IMSL routines is in computing some of the spline function interpolants. In certain cases the function we are fitting with a spline change rapidly, typically as $x \to \infty$ or $\alpha \uparrow 2$ or $\alpha \downarrow 0$. Even though the function values are calculated accurately, a spline computed by the traditional methods introduce oscillations in the approximation in the region where the function changes rapidly. We have had to resort to using a special spline routine DCSCON from the IMSL libraries. This routine computes a 'shape preserving' spline that follows the concavity of the data. This solves one of the problems, but slows things down a bit, and it relies on proprietary routines that don't always converge. There has recently been some new work on this problem, see Dontchev

et al. (2002), which claims to have a globally convergent method. We have been in contact with those authors, and are considering coding their improvements and obtaining a non-proprietary method.

There is still work to be done to make these routines faster and use less memory. This is especially true if these methods are to be used in a real-time DSP environment. For such an application, methods like the rational approximation to the nonlinear function, called g_3 above, are probably necessary. In such applications it is probably reasonable to restrict the parameter space to $1 < \alpha \leq 2$ and $\beta = 0$, or at least $|\beta| << 1$. It is likely not important to get high precision in all calculations, but is important to have a reasonable approximation that captures the key features of the quantity of interest.

We caution that the analysis done in this contract were based on simulated stable distributions. It is not clear how well these methods will work with real data. It is hoped that these methods, while not giving an exact fit to the data, give a better fit than a Gaussian distribution. Discrepancies from an exact stable distribution may not be important if an approximate fit with a stable model gives practical, robust methods for working with radar and other signals.

3 Miscellaneous

A summary CD-ROM is being delivered to the TPOCs. That CD contains: the contract proposal, the agreed upon work plan, the 6 monthly reports, this report, the matlab interface to STABLE, and the user manual for the STABLE routines.

The work described, particularly the matlab interface, took more time than expected. A subcontractor, Dr. Alex White, was hired to implement initial versions of some of the computational and graphical routines and to run and analyze the simulations of the estimation methods. He did 320 hours of work. The PI put in more work than planned, because of the reasons mentioned above. Significant time was spent on the matlab interface, on the implementation of the discrete maximum likelihood method of estimating stable parameters, on the fast approximation of the cdf and on the nonlinear function. The work for Dr. John Nolan totaled 550 hours.

Finally, we comment briefly on commercialization of this work. We have announced the availability of the STABLE interface on our website[1], and have three copies of the matlab interface and one Mathematica interface. Recently MathWorks, the company that sells matlab, accepted Robust Analysis as a "Connection Partner", and lists STABLE on their website[2]. The PI has used an S-Plus interface

[1] `http://www.RobustAnalysis.com`

[2] `http://www.mathtools.net/MATLAB/Add-on_functions/index.html`

for doing the development and testing of many routines and we are working with Insightful, the maker of S-Plus, on a Phase II proposal to continue this work.

References

Dontchev, A. L., H. D. Qi, L. Qi, and H. Yin (2002). A Newton method for shape-preserving spline interpolation. *SIAM J. Optimization 13*, 588–602.

Fama, E. and R. Roll (1968). Some properties of symmetric stable distributions. *JASA 63*, 817–83.

Kogon, S. M. and D. B. Williams (1998). Characteristic function based estimation of stable parameters. In R. Adler, R. Feldman, and M. Taqqu (Eds.), *A Practical Guide to Heavy Tailed Data*, pp. 311–338. Boston, MA.: Birkhäuser.

Koutrouvelis, I. A. (1980). Regression type estimation of the parameters of stable laws. *JASA 75*, 918–928.

McCulloch, J. H. (1986). Simple consistent estimators of stable distribution parameters. *Communications in Statistics. Simulation and Computation 15*, 1109–1136.

Nolan, J. P. (1997). Numerical calculation of stable densities and distribution functions. *Commun. Statist. -Stochastic Models 13*, 759–774.

Nolan, J. P. (2000). Maximum likelihood estimation of stable parameters. In O. E. Barndorff-Nielsen, T. Mikosch, and S. I. Resnick (Eds.), *Lévy Processes: Theory and Applications*. Boston: Birkhäuser.

Ojeda, D. (2001). *Comparative study of stable parameter estimators and regression with stably distributed errors*. Ph. D. thesis, American University.

Zolotarev, V. M. (1986). *One-dimensional Stable Distributions*, Volume 65 of Translations of mathematical monographs. American Mathematical Society. Translation from the original 1983 Russian edition.